Geekbrains

**Разработка чат-бота для знакомства в социальной сети Telegram**

Специализация Backend-разработчик

Силантьев Леонид Алексеевич

г. Санкт-Петербург

2024

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 4

ГЛАВА 1. Теоретические основы 6

* 1. Мессенджеры и их разновидности 6
  2. Чат-боты и их различия 10
  3. Обзор аналогов 12
     1. Чат-бот Дайвинчик 12
     2. Чат-бот Camilfo Club 13
     3. Чат-бот Анонимные знакомства 14
     4. Чат-бот Чат рулетка 15

ГЛАВА 2. Выбор технологий и среды разработки 17

* 1. Выбранные технологии и среды разработки 17
     1. Aiogram 17
     2. База данных MySQL 18
     3. Python 19
     4. VPC/VDS - Виртуальный сервер 21
        1. Технология визуализации OpenVZ и KVM 22
  2. Среда разработки 23
     1. Visual Studio Code 23

ГЛАВА 3. Разработка Telegram бота 25

* 1. Описание функционала чат-бота для знакомств 25
  2. Анализ целевой аудитории 26
  3. Проектирование структуры и интерфейса чат-бота 26
  4. Регистрация чат-бота 28
  5. Реализация клиентской части чат-бота 31
     1. Разработка, создание и обработка 31
     2. Асинхронное выполнение, asyncio 35
     3. FSM – машинное состояние 37
     4. Создание Анкеты и работа связанных функций 41
  6. Проектирование и реализация базы данных чат-бота 45
  7. Реализация серверной части чат-бота 48
     1. Реализация магазина 51
  8. Размещение чат-бота на удаленном сервере 53

ЗАКЛЮЧЕНИЕ 56

Список используемых источников 57

Приложения 59

**ВВЕДЕНИЕ**

В нашем современном мире, пронизанном технологиями, социальные сети играют ключевую роль в нашей повседневной жизни. Они стали не просто средством общения, но и площадкой для знакомства, обмена информацией, и даже поиска потенциальных партнеров. В этом контексте, разработка инновационных подходов к созданию сервисов знакомств становится чрезвычайно важной задачей.

Telegram, один из наиболее популярных мессенджеров, предоставляет уникальные возможности для разработки и внедрения различных сервисов, в том числе и для знакомств. С его открытым API, широким функционалом и высоким уровнем безопасности, Telegram становится идеальной платформой для создания чат-ботов, способных решать разнообразные задачи, информационные каналы, группы для общения, с недавнего времени, создание игр и ключевое для общества облегчение процесса знакомства.

Данный дипломный проект разрабатывался на основе специальности Backend-разработчика.

Цель данной дипломной работы заключается в разработке чат-бота для знакомства в социальной сети Telegram. Наша цель - создать инновационный сервис, который поможет людям находить новых друзей, партнеров по интересам, или даже возможных романтических партнеров, используя возможности Telegram, встроенных функций и сторонних библиотек, и их возможностей.

Это введение проведет нас через актуальность выбранной темы, определяя основные проблемы, которые предстоит решить, и обозначит основные этапы работы по разработке чат-бота для знакомств в Telegram. Мы также рассмотрим актуальные вызовы, стоящие перед разработчиками сервисов знакомств, и возможные пути их решения с использованием передовых технологий.

Исходя из поставленной цели, были поставлены следующие задачи:

1. анализ выбранной предметной области;
2. сравнение имеющихся аналогов чат-ботов;
3. выбор технологий и среды разработки;
4. разработка чат-бота на платформе Telegram.

**ГЛАВА 1**

**Теоретические основы**

* 1. **Мессенджеры и их разновидности.**

Свое развитие интернет-сервисы для общения начали с чатов, потом мессенджеры, потом социальные сети, но с недавних пор мессенджеры вновь возглавили список самых перспективных сервисов. Это подтверждаю исследования компании «Exploding Topics» в 2024 году: количество пользователей в мессенджерах в мире увеличилось по экспоненте, а именно приблизительно около 3,32 миллиардов челок используют мессенджеры. Одновременно с этим среднее количество мессенджеров, используемых одним пользователем, увеличилось за 2017 год с 1,73 до 3 штук. Последние данные показывают, что ежемесячно насчитывается около 571 миллиона активных пользователей QQ. Мессенджеры используются преимущественно взрослыми, так на долю лиц до 18 лет приходится 7,7% от общего количества аудитории, пользователи в возрасте от 18 до 25 лет составляют 9,8%, от 25 до 35 лет – 32,3%, от 35 до 45 лет – 26,8%. При этом 13,1% пользователей мессенджеров находятся в возрасте от 45 до 55 лет, а 7,6% — от 55 до 64 лет. И всего 2,7% аудитории мессенджеров находится в возрасте старше 64 лет.

Причина волны популярности мессенджеров –

* Мобильные устройства дешевеют, их вычислительная мощность растет. За последние 7 лет стоимость смартфонов заметно снизилась, а их характеристики, наоборот, улучшились.
* Миграция пользователей в мобильные сети и рост числа активных пользователей мобильного широкополосного доступа. Статистика, опубликованная в конце прошлого года Международным союзом электросвязи, гласит, что в мире уже 6,8 млрд. пользователей мобильной связи, что примерно эквивалентно 96% населения земли.
* Мобильный трафик перетекает в интернет-мессенджеры. То, что мобильный трафик перетекает в интернет-мессенджеры, доказал пример Facebook и WhatsApp: будучи неспособной сколь-нибудь существенно нарастить аудиторию, Facebook купила мессенджер с полумиллиардом пользователей для того, чтобы «обратить их в свою веру». К слову, убытки от работы WhatsApp за 2013 год операторы сотовой связи оценили в $32,5 млн, а к 2016 году эти убытки возрастут до $54 млн.

В настоящее время существует большое разнообразие мессенджеров,

среди которых стоит отметить некоторые из них, а также их преимущества и

недостатки (Приложение А).

Особого внимания заслуживает мессенджер Telegram.

Telegram – это кроссплатформенное приложение, разработанное на языке программирования C++, позволяющее обмениваться сообщениями и файлами большинства форматов. Мессенджер использует специально разработанную серверную часть с закрытым кодом, работающим на серверах Германии, США, ОАЭ. Telegram имеет ряд преимуществ:

* приватность – все чаты зашифровываются, а сообщения

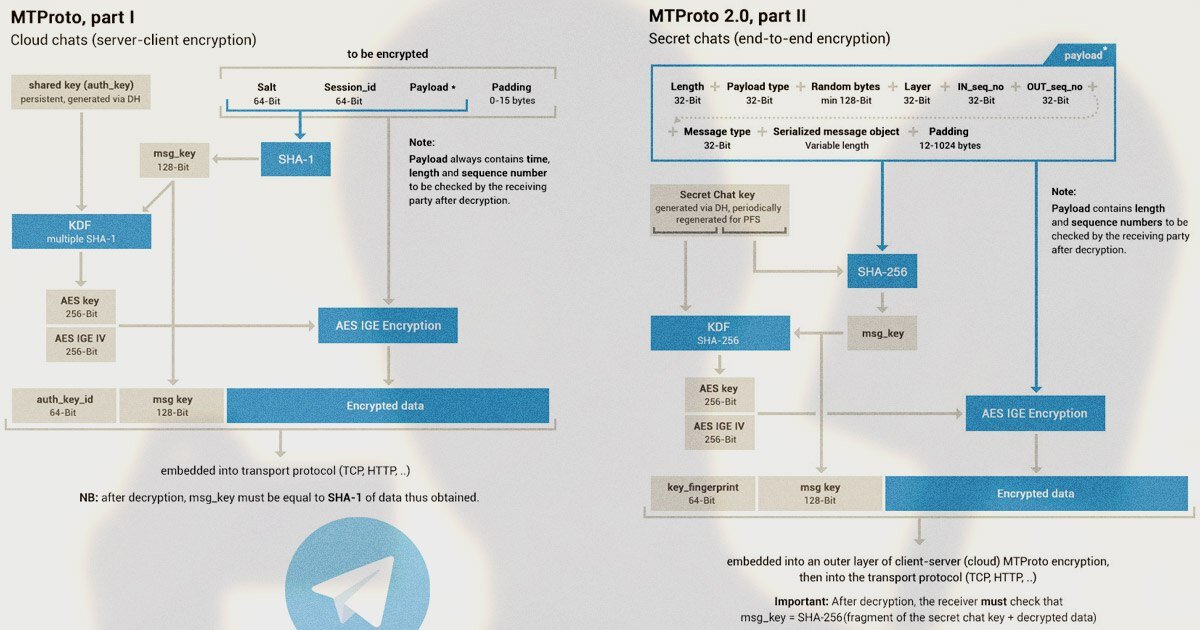
уничтожаются через обозначенное время;

* быстрота – скорость доставки сообщений выше, чем у аналогов;
* распределение – сервера Telegram расположены по всему миру, что

повышает отказоустойчивость;

* открытость – использование открытого протокола MTProto и API,

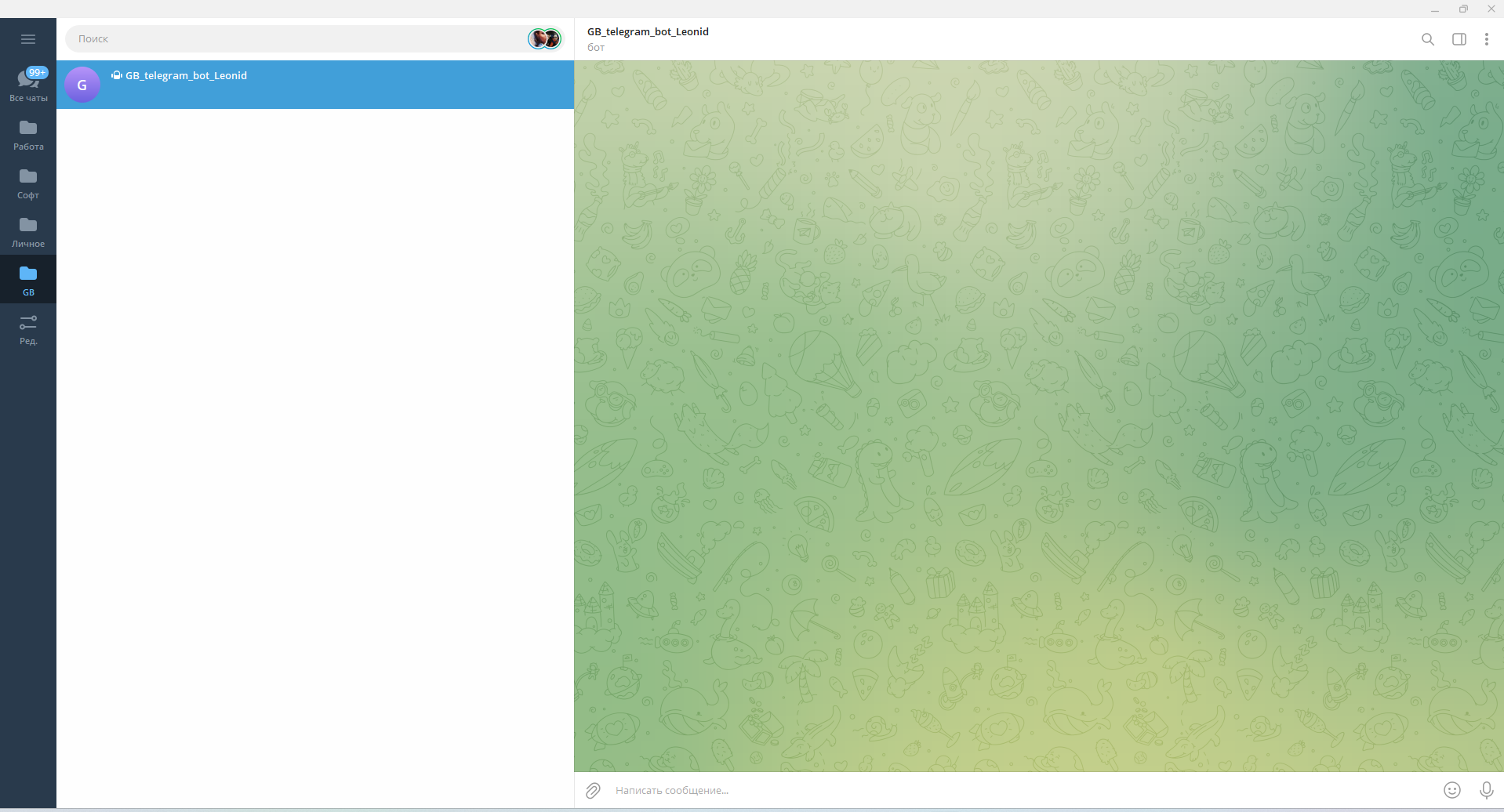
бесплатных для всех; Основателями мессенджера Telegram являются братья Павел и Николай Дуровы. Павел является финансовой и идеологической опорой проекта, а Николай занят техническими аспектами. Особенностями данного мессенджера являются секретные чаты и специально разработанный протокол шифрования MTProto (рисунок 1)



Протокол шифрования MTProto

Секретные чаты нужны для людей, которые хотят обеспечить себя максимально безопасным общением. Помимо использования основного протокола все сообщения шифруются ключами по принципу от устройства до устройства. Никто не перехватит и не расшифрует сообщения, включая сотрудников Telegram. Сообщения из таких чатов пересылать нельзя и они не хранятся на серверах приложения. Но у секретных чатов есть два недостатка – это невозможность открыть чат на устройстве, которое не участвовало в его создании, и никто не защищен от скриншота интерфейса Telegram с отрытым секретным чатом.

Мессенджер Telegram доступен для всех популярных платформ, например, Android OS, iOS, Windows Phone, Linux и другие. Так же в большинстве версия для разных операционных систем доступны как минимум английский, русский, польский и немецкий языки. Примеры интерфейса десктопного приложения на Windows 11 и Linux изображены на рисунках 2 и 3.



(интерфейса десктопного приложения на Windows 11 рисунок 2)

Также в большинстве версия для разных операционных систем доступны как минимум английский, русский, польский и немецкий языки и многие дургие.



(интерфейса десктопного приложения на Linux рисунок 3)

В последнее время повышается интерес к блокчейн-платформе Telegram Open Network и нараставшей популярности криптовалюты TON.

* 1. **Чат-боты**

Чат-бот — это программа, способная имитировать разговор с человеком, используя инструменты обработки естественного языка (NLP), такие как сопоставления с образцом (pattern matching) и большие языковые модели (LLM).

Упрощённо работу чат-бота можно свести к четырём шагам:

* Пользователь отправляет ему запрос.
* Чат-бот анализирует запрос.
* Чат-бот генерирует ответ.
* Отправляет ответ пользователю.

Алан Тьюринг — человек, с которого всё началось. В 1950 году Алан Тьюринг, пионер компьютеров, написал научную статью под названием «Вычислительные машины и интеллект». В статье ученый имел в виду, что компьютерная программа может думать и говорить как человек. Чтобы доказать это, Тьюринг предложил эксперимент под названием «Имитационная игра», который сегодня известен как тест Тьюринга.

В 1966 году профессор Массачусетского технологического института Джозеф Вейценбаум разработал компьютерную программу под названием Элиза. Считается, что это первый чат-бот в истории. Элиза была простым чат-ботом на основе ключевых слов, имитирующим человека-психиатра. Программа общалась, сопоставляя вопросы пользователей со сценариями ответов, введенными в ее базу данных.

Когда пациент говорил: «Моя мама любит цветы», Элиза отвечала: «Расскажи мне больше о своей матери». Таким образом Элиза побуждала людей больше говорить. Это также создало впечатление, что она понимает человеческую речь так же, как люди. Конечно, это было только впечатление понимания. Создание Элизы дало толчок развитию обработки естественного языка.

Боты – это программы, которые выполняют различные задачи для пользователя, находящегося в мессенджере. Бот выглядит как обычный чат, однако общение происходит не с человеком, а с программой, которая может принять заказ на вызов машины, если это – бот такси, или прислать свежие статьи, если это – новостной бот, или заказать доставку еды на дом, если это – ресторанный бот. Не так давно чат-боты обрели большую популярность, перевоплотившись из развлечения в более серьезную вещь, так как они, в основном, стали использоваться для решения серьезных бизнес-задач. В эпоху информационных технологий – это нормально явление, а тем более – сети Интернет, ведь общество давно перешло на «новое» и деловое, и неделовое общение. Во-первых, чат-боты – это «платформы» для решения бизнес-задач. Во-вторых, чат-бот – это приложение, которое поддерживает диалог с пользователем, выбирая ответы из базы данных: вы спрашиваете, где пообедать и тут же получаете мгновенный ответ. Кроме того, чат-боты выполняют множество полезных функций по исполнению рутинных операций, поиску информации, объединению данных, работе с клиентурой.

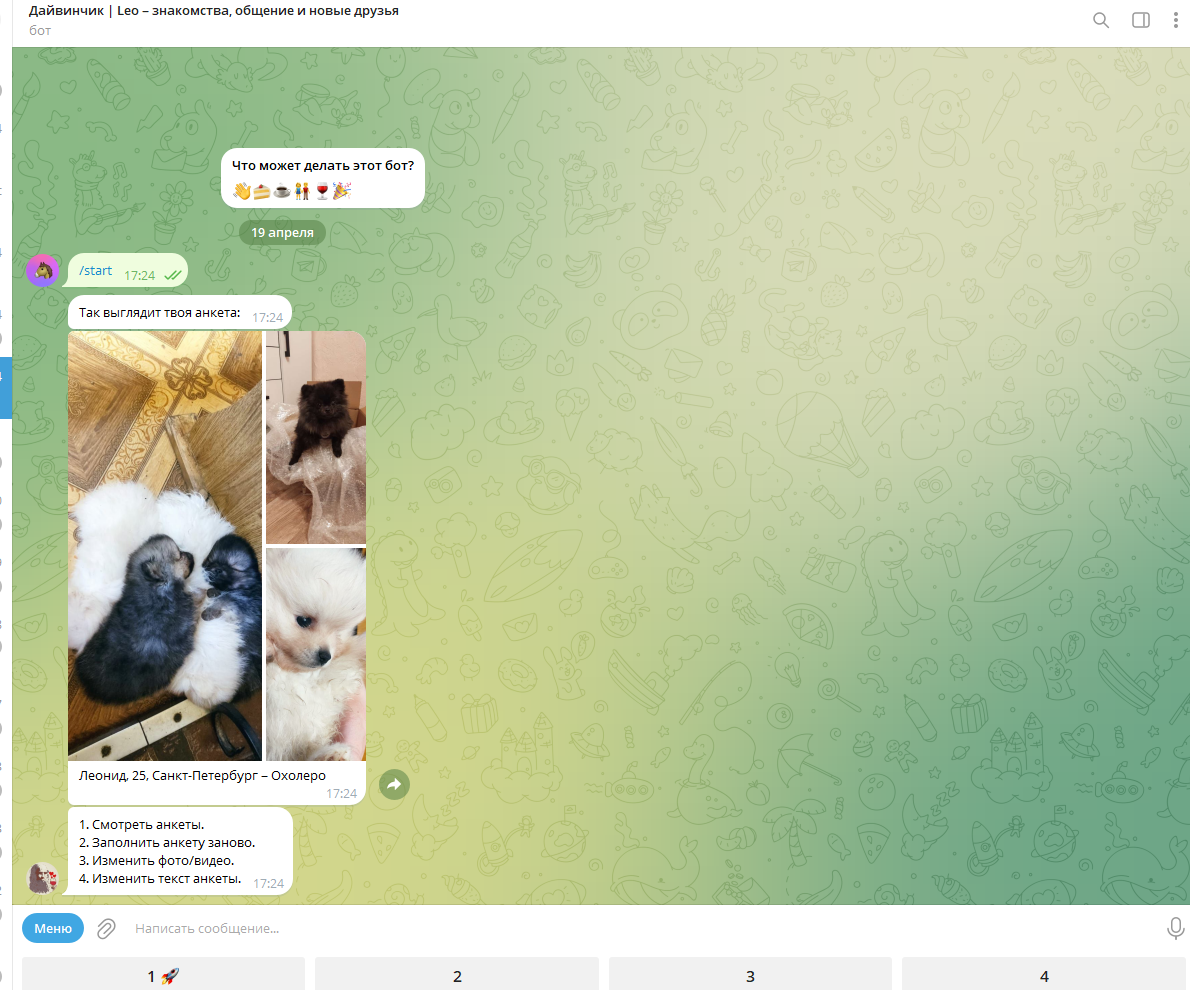
Чат-бот как виртуальный собеседник имеет базу знаний, которая представляет собой наборы возможных вопросов пользователя и соответствующих им ответов. Наиболее распространенными вариантами для получения нужного ответа являются ключевые слова, совпадение фразы, совпадение контекста. Всегда существуют какие-то простые и легкие по выполнению дела, на которые не хочется тратить время. Тут на помощь всегда могут прийти чат-боты. Век пишущих машинок, ходьбы по библиотекам и бесконечных очередей в книжные магазины канул в Лету. Теперь для сбора 11 информации можно использовать чат-ботов. Конечно, это делается при помощи общения с людьми, но отнюдь не всегда. Например, в рамках какого-либо мероприятия чат-боты могут сообщать всем участникам новости и предоставлять справочную информацию.

* 1. **Чат-боты**
     1. Чат-бот Дайвинчик

Согласно сайту (*vc.ru)*, и многим другим сайтам связанных поисковым запросом “чат бот для знакомств в телеграмм”, Дайвинчик является лидирующим среди остальных. Зарождение бота произошло на волне популярности группы с таким же названием в сот-сети ВКонтакте. Цель бота состояла в том, чтоб захватить больше пользователей в различных мессенджерах, особенно после появления Telegram. Аналогичный бот в других платформ не сыскал такой популярности как в вышеупомянутой платформе по тем же причинам, почему Telegram популярен и по нынешний день.

Работоспособность оставляет желать лучшего, бот работает исправно, но важные, ключевые возможности либо отсутствуют, либо работают неисправно. Интерфейс сделан хорошо, его простая эргономика и возможности делают его “user-friendly”.

Из-за огромной аудитории в различных мессенджерах Дайвинчик все еще популярен и продолжает работу в частичном исправном состоянии. Интерфейс чат-бота Window 11 изображен на рисунке 4.

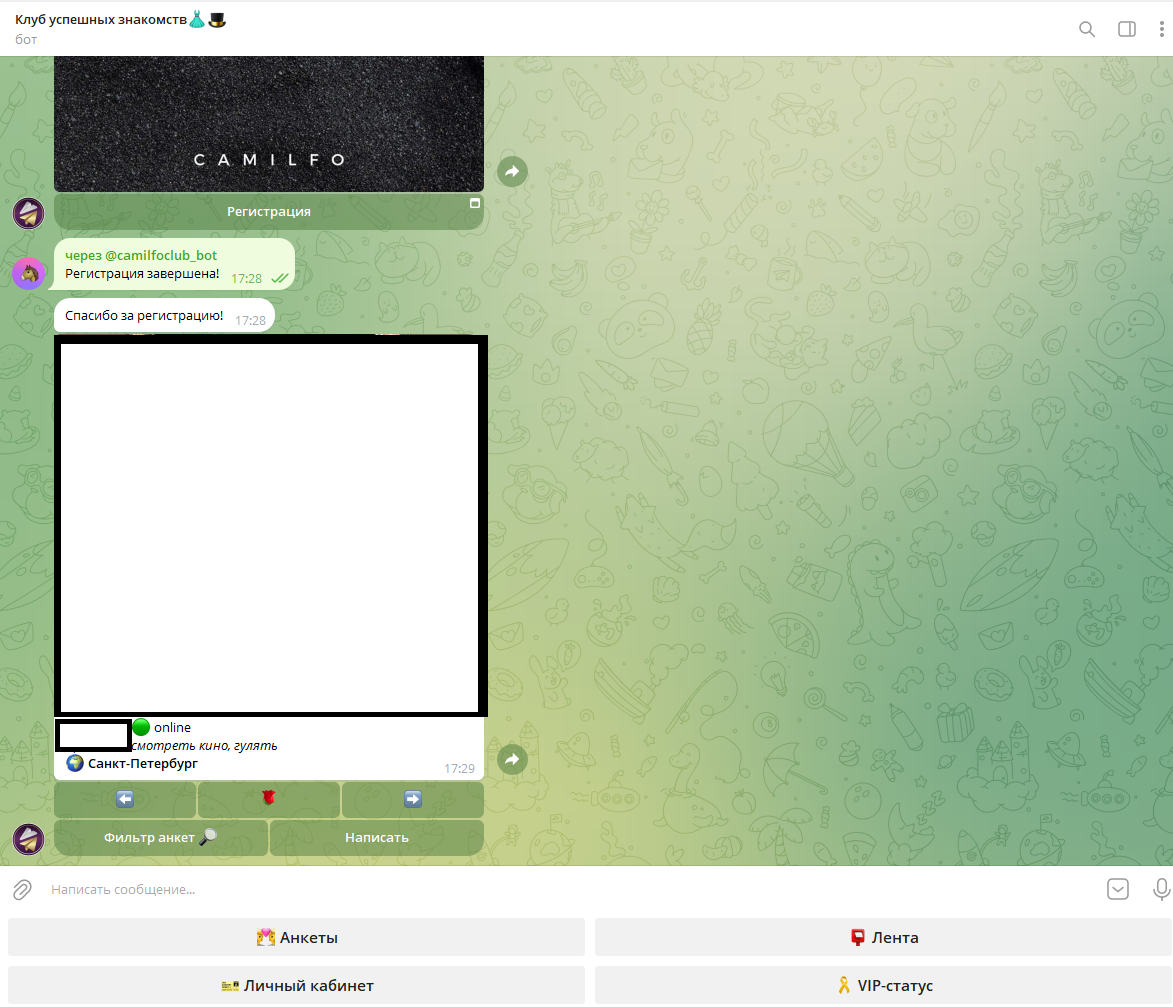
****

(Интерфейс чат-бота Дайвинчик рисунок 4.)

* + 1. Чат-бот Camilfo Club

В случае с чат-ботом Camilfo Club, информацию, которую находишь в интернете по поиску места для знакомств в вышеупомянутом боте, ставишь под сомнение. Дайвинчик хоть и является лидером в списках, но его уверена может сместить Camilfo Club.

Функционал, интерфейс, работа, все работает исправно, по сравнению с конкурентом выше, можно сказать, что все работает, но заложенный функционал перегружен, частично неисправен Online зарегистрированного пользователя, отсутствует автономность все зарегистрированные пользователи проходят модерацию в очень длинные сроки.

По количеству пользователей Camilfo Club значительно уступает чат-боту Дайвинчика, по естественным причинам. Интерфейс чат-бота Window 11 изображен на рисунке 5.

(Интерфейс чат-бота Camilfo Club рисунок 5.)

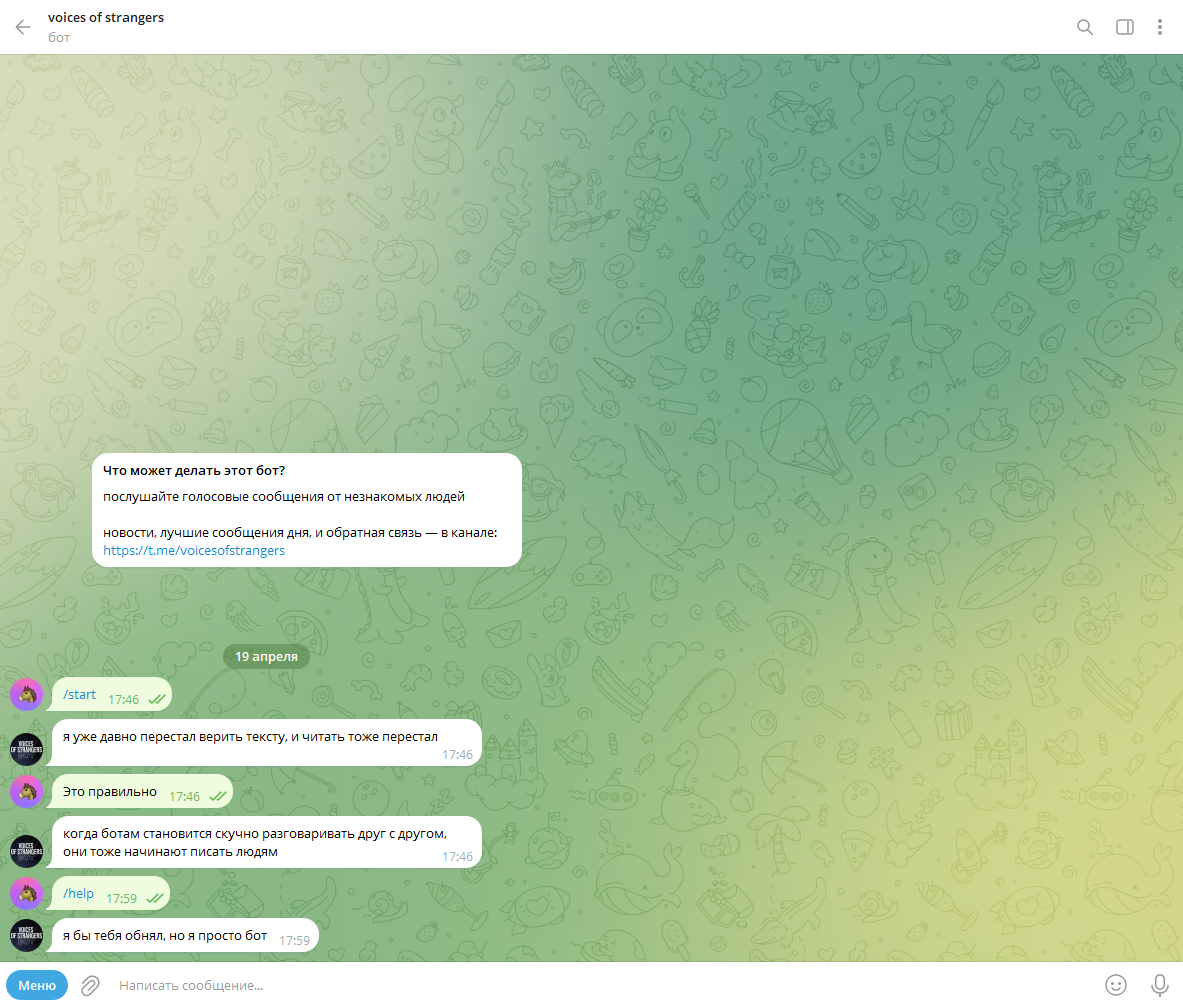
* + 1. Чат-бот Анонимные знакомства

На данный момент по обширному интерфейсу, объему пользователей и функционалу работы чат-боты заканчиваются, на их место приходят новые, быстро развивающиеся и постоянно проектируемые чат-боты аналогичные “ Чат-бот Анонимные знакомства”.

Из-за отсутствия большего функционала, интерфейс минимальный, что приводит нас к “not user friendly” общение с чат-ботом в большинстве случаев сводится к стандартным командам, а именно:

* /start
* /help
* /vip

Простота, функционал и различные конструкторы по созданию Telegam-ботов, заполнили Telegram большим количеством одинаковых аналогов, что приводит к низким показателем вовлеченности и снижает количество пользователей в этих чат-ботах. Интерфейс чат-бота Window 11 изображен на рисунке 4.



(Интерфейс чат-бота Анонимные знакомства рисунок 6.)

* + 1. Чат-бот Чат рулетка

Что касается “Чат рулеток”, пользуюсь активным ростом Telegam и большим разнообразием контента, весь набор от общения до контента 18+, соизмеримо с ростом вышеупомянутым мессенджером, быстро набрал популярность и занял соответствующие позиции в поисковых выдачах.

Как и в пункте выше, большая часть чат-ботов в этой категории не имеет хорошего функционала и удобства использования, но при это по сравнению с “чат-бота Анонимные знакомства”, отсутствие интерфейса с лихвой компенсируется контентом в этой нише.

Именно чат рулетки заложили основной функционал связанный с возможностью оплатить подписку, это есть и у конкурентов описанных выше, но именно чат рулетки зачастую пользуются схемой с невозможным отключением этой самой подписки. Интерфейс чат-бота Window 11 изображен на рисунке 7.



(Интерфейс Чат рулетка рисунок 7.)

**ГЛАВА 2**

**Выбор технологий и среды разработки**

* 1. Выбранные технологии разработки
     1. Aiogram

Aiogram. Современная библиотека, набирающая популярность: многие чат-боты написаны на ней. В этой и последующих статьях цикла мы будем работать именно с Aiogram. Библиотека реализует асинхронное выполнение кода, что позволяет не останавливать работу бота в ожидании ответа пользователя. Кроме того, у Aiogram есть подробная документация и большое русскоязычное комьюнити.

Как уже было сказано ранее, одно из главных достоинств библиотеки — полная асинхронность. Она использует синтаксис async/await, который позволяет программе выполнять несколько задач одновременно и эффективно управлять потоком выполнения.

Вот простой пример функции, использующей механизм async/await:

Функция, которая обрабатывает сообщение пользователя в Telegram, называется хендлером, то есть обработчиком. У каждой команды или группы команд свой обработчик.

# Хендлер для команды /start

@dp.message(Command("start"))

async def start(message: types.Message):

await message.answer("Привет, пользователь!")

В хендлере мы прописываем, что бот должен сделать в ответ на сообщение. А для того, чтобы для каждой команды вызывался нужный обработчик, функция оборачивается в декоратор, которому передаётся имя команды без символа /.

Служебное слово async указывает интерпретатору, что функция будет работать в асинхронном режиме. Это означает, что интерпретатору не нужно ждать, пока выполняется код функции, — он может выполнять следующие инструкции, пока start что-нибудь не вернёт. Это «что-нибудь» следует за служебным словом await («ожидать»), а не return, как в обычном коде.

Другой плюс Aiogram — в большом наборе инструментов и хуков, которые можно использовать для добавления дополнительных функций и настроек бота. Библиотека обеспечивает полный доступ ко всем возможностям Telegram API, включая отправку и получение сообщений, управление клавиатурой, обработку медиафайлов (фотографий, видео, документов) и многое другое.

* + 1. База данных MySQL

MySQL — это реляционная база данных (СУБД). Реляционная — значит внутри неё есть данные, которые связаны между собой, и эту связь можно представить в виде таблиц.

Технически MySQL — это много таблиц, как-то связанных между собой. Например, одна отвечает за товары, другая — за покупки, третья — за клиентов.

Разработчиком MySQL, популярной SQL-базы данных с открытым кодом, является компания MySQL AB. В настоящее время компания куплена корпорцией Oracle, которой и принадлежит теперь продукт. Однако MySQL по-прежнему остается базой данных с открытым кодом. Свое происхождение MySQL ведет от продукта mSQL, разработанного в конце 1970-х гг. компанией TcX и использовавшемуся для доступа к таблицам, для которых использовались собственные быстрые подпрограммы низкого уровня. Однако после тестирования был сделан вывод, что скорость и гибкость mSQL недостаточны. В результате для базы данных был разработан новый SQL-интерфейс. Новый продукт получил название MySQL. Массовое же признание MySQL получила начиная с линейки продуктов версии 3, которые стали широко использоваться на серверах в сети Интернет. В настоящее время используется 5 версия продукта.

Ниже приведено описание важных характеристик программного обеспечения MySQL:

* Внутренние характеристики и переносимость
* Написан на C и C++. Протестирован на множестве различных компиляторов.
* Работает на различных аппаратных платформах и разных операционных системах.
* Высокая производительность за счет максимально оптимизированного кода, эффективной системы распределения памяти и продуманной системы дисковых таблиц.
* Безопасность
* Система, основанная на привилегиях и паролях, за счет чего обеспечивается гибкость и безопасность, и с возможностью верификации с удаленного компьютера. Пароли защищены, т.к. они при передаче по сети при соединении с сервером шифруются.
* Масштабируемость
* Способность работать с очень большими базами данных (десятки и сотни миллионов записей).
* Возможность кластеризации серверов и распределения обработки информации между серверами
  + 1. Python

Язык Python является одним из самых простых в изучении и самых приятных в использовании из языков программирования, получивших широкое распространение. Программный код на языке Python легко читать и писать, и, будучи лаконичным, он не выглядит загадочным. Python - очень выразительный язык, позволяющий уместить приложение в меньшее количество строк, чем на это потребовалось бы в других языках, таких как C++ или Java.

Python — это высокоуровневый язык программирования, отличающийся эффективностью, простотой и универсальностью использования. Он широко применяется в разработке веб-приложений и прикладного программного обеспечения, а также в машинном обучении и обработке больших данных. За счет простого и интуитивно понятного синтаксиса является одним из распространенных языков для обучения программированию.

Для чего нужен Python:

* Data Science и машинное обучение. Эти два направления IT тесно связаны друг с другом. Наука о данных заключается в обработке больших массивов информации из базы данных, а машинное обучение — в разработке компьютерных алгоритмов, способных учиться на ней и делать точные прогнозы. В Data Science используют Python для включения очистки и разметки данных, поиска и обработки статистической информации, ее визуализацию в виде диаграмм, графиков и т.д. С помощью библиотеки Python ML классифицируются изображения, тексты, поисковый трафик, осуществляется распознавание лиц и речи, глубинное машинное обучение.
* Веб-разработка. Многие крупные интернет-компании, такие как Google, Facebook\*, программируют на Python свои самые известные проекты, например, Instagram\*, YouTube, Dropbox и т.д. Этот язык позволяет вести веб-разработку на стороне сервера, потому что его обширная библиотека включает множество решений как раз для реализации сложных серверных функций. За счет своей простоты использования Python широко применяется небольшими командами и одиночными разработчиками для создания сайтов, десктопных и мобильных веб-приложений.
* Разработка и тестирование ПО. Возможности Python используются тестировщиками и разработчиками для поиска и исправления ошибок, автоматической сборки, разработки прототипов программного обеспечения, управления проектами и т.д. Кроме того, с помощью сред модульного тестирования «Питона» осуществляется проверка функций. Также на этом языке создаются тестовые скрипты, имитирующие различные сценарии использования ПО. Разработчики аппаратных платформ (например, IBM, Hewlett-Packard, Intel) тоже используют Python для тестирования своей продукции.
  + 1. VPC/VDS - Виртуальный сервер

VDS (Virtual Dedicated Server) или VPS (Virtual Private Server) — это хостинг-услуга, где пользователю предоставляется виртуальный сервер с максимальными привилегиями. VDS или VPS эмулирует работу реального физического сервера — есть root-доступ, возможна установка своих операционных систем и программного обеспечения. На одном физическом сервере обычно работает несколько независимых виртуальных серверов.

Бытует мнение, что термины обозначают виртуальные серверы с различными типами виртуализации: VPS обозначает виртуализацию на уровне операционной системы, VDS — аппаратную виртуализацию. На самом деле, оба термина появились и развивались параллельно, и обозначают одно и то же: виртуальный выделенный сервер, запущенный на базе физического.

По принципу работы и особенностям функционирования виртуальный выделенный сервер мало чем отличается от физического сервера. Он не имеет ограничений по количеству размещаемых сайтов, баз данных, доменных зон, ssh- ftp- email-пользователей, а его владелец-администратор имеет право работать с файлами, устанавливать приложения и совершать другие операции, как на реальном полноценном сервере. Каждый виртуальный сервер на нашем хостинге имеет собственный IP-адрес, который не разделяется с кем-либо еще. При необходимости можно заказывать дополнительные IP-адреса.

Виртуальный сервер работает независимо от других, расположенных на одной с ним хост-машине. Это значит, что сбой в работе одного сервера не влияет на функционирование «соседей».

В отличие от обычного виртуального (shared) хостинга, на виртуальном сервере доступны технологии защиты от DDoS-атак, а сохранность данных гарантируется возможностью создания резервной копии.

* + - 1. Технология визуализации OpenVZ и KVM

Технология OpenVZ базируется на ядре ОС Linux и позволяет на одном физическом сервере создавать и запускать изолированные друг от друга копии операционной системы — так называемые «виртуальные частные серверы» (Virtual Private Servers, VPS) или «виртуальные среды» (Virtual Environments, VE). Данная технология отличается высокой производительностью, простотой использования и легкостью управления сервером. При этом выбор операционной системы возможен только из ряда предложенных (Debian, CentOS, Ubuntu), так как виртуальные серверы конкретной хост-машины используют общее ядро Linux. Смена тарифа и изменение ресурсов не требует перезагрузки.

KVM (Kernel-based Virtual Machine) – технология аппаратной виртуализации, позволяющая создать на хост-машине полный виртуальный аналог реального физического сервера. Каждому такому серверу выделяется своя область в оперативной памяти и пространство на жестком диске, собственная сетевая карта, что повышает общую надежность работы такого сервера. Фактически технология KVM позволяет создать полностью изолированный от «соседей» виртуальный сервер с собственным ядром ОС, который пользователь может настраивать и модифицировать под собственные нужды практически без ограничений. Возможна установка любой операционной системы: Linux, FreeBSD, Windows и даже собственного образа. Смена тарифа и изменение ресурсов только после перезагрузки.

* 1. Среда разработки
     1. Visual Studio Code

Visual Studio Code (VS Code) — это кросс-платформенный редактор кода от компании Microsoft, разработанный на базе фреймворка Electron. С его помощью можно разрабатывать кросс-платформенные десктопные приложения, используя веб-технологии.

Суперсила VS Code в том, что он не привязан к определённому языку программирования, поэтому с его помощью можно создавать сайты, мобильные приложения, работать с базами данных и тестировать сервисы. Огромная библиотека плагинов позволяет расширять функции редактора, а если не удастся найти подходящий плагин, то всегда можно написать свой.

Возможности VS Code:

* Подсветка синтаксиса — функции, классы, переменные и другие сущности выделяются разными цветами.
* Автоматическое дополнение — если начать что-то писать, редактор предложит варианты завершения строки.
* Контроль версий — поддерживаются интеграции с GitHub, GitLens и другими похожими сервисами.
* Отладка — редактор подсвечивает ошибки и предлагает исправления. Вместе с этим поддерживается и полноценный режим отладки кода. К примеру, для проекта на Python его можно запустить клавишей F5, а для JavaScript запустится новое окно браузера.
* Рефакторинг — редактор кода выводит советы для улучшения кода и повышения производительности, подсказывая, какие конструкции можно заменить.

**ГЛАВА 3**

**Разработка Telegram бота**

* 1. Описание функционала чат-бота для знакомств**.**

Получив достаточную информацию, связанную с чат-ботами конкурентов, оценив их минусы и плюсы, выстраивается картина собственного проекта. Основные функции:

* Создание и регистрация анкеты для пользователя и ее дальнейшая автономность, не повторяя ошибки конкурентов.
* Анкета пользователя должна быть “User Friendly”, это значит, что интерфейс взаимодействия с функционалом и возможностями бота должен быть понятен для новых пользователей.
* Анкета не должна быть перегружена, но и не должна быть слишком простой по заполнению. Анкета будет состоять из следующих пунктов:

1. Имя и если, есть, то и Фамилия, а в дальнейшем именующие себя как first\_name и last\_name.
2. Анкета должна состоять минимум из 3 фотографий, и, если у пользователя появится желание оставить 1-у или 2-е фотографии, реализовать соответствующий функционал.
3. Возраст, ключевой элемент в анкете чат-ботов для знакомств, должен присутствовать в анкете в ярком и **жирном** виде.
4. Описание критический важный элемент в анкете, по сравнению с конкурентами, в нашем проекте ограничение по количеству символов будет составлять 20 символов.
5. Место расположение, элемент такой-же критический, как и описание, должен быть виден хорошо и заметно

* Реализовать правильную работу поиска анкет.
* После совпадений интересов, а именно после обоюдной оценки анкеты, появляется возможность перейти по ссылке, к конкретному пользователю.
* Уведомление, важный элемент в анкете, после определенного времени, не после каждой прожатой кнопки в интерфейсе, которая отвечает за нравится, будет приходить уведомление о количестве человек заинтересовавшихся анкетой, если таковы есть.
  1. Анализ целевой аудитории**.**

Как и в описанных плюсах и нюансов конкурентов в темах выше, целевая аудитория именующиеся в дальнейшем, как ЦА, будет иметь возраст в среднем от 18 до 38 лет. Большая часть ЦА из-за отсутствия разнообразия и выбора в данной нише, будут в поиске натыкаться на наш проект. В данном случае, мы понимаем, что новые пользователи уже имели опыт и понимание в пользовании ботами в Telegram, из чего следует, что при проектировании и реализации интерфейса и функционала нашего чат-бота, нужно обращаться к возможностям, названиям и расположению интерфейса с кнопками, в таком же исполнении, как и у конкурентов, но, для того чтобы наш проект выделился среди конкурентов, необходимо, правильно подойти к отображению и группировке, располагаемого функционала. В случае, если новые пользователи не имели опыта в использовании чат-ботов где-либо угодно, нужно написать и расписать возможный функционал при использовании, в отдельных сообщениях, для того, чтоб новые пользователи смогли привыкнуть и остаться удовлетворёнными в тех целях, которые себе ставит ЦА в чат-ботах для знакомств.

* 1. Проектирование структуры и интерфейса чат-бота.

После проделанной работы в анализе рынка, ниш, ЦА начинается формироваться картина проекта. За основу отображения возможностях чат-бота, возьмем отображения чат-бота Дайвинчик. Выбор основан на количестве пользователей и крайне простым и одновременно информативным отображением возможностей.

Сформируем по пунктам, от приветствия до отображения уведомления, интерфейс проекта. Приветствие, то, что встречает нас, при переходе к боту. При наличии анкеты в чат-боте Дайвинчик, бот присылает зарегистрированную анкету, 4 кнопки и текстовую подсказку. Возьмем в качестве шаблона именно такой формат, со своими доработками.

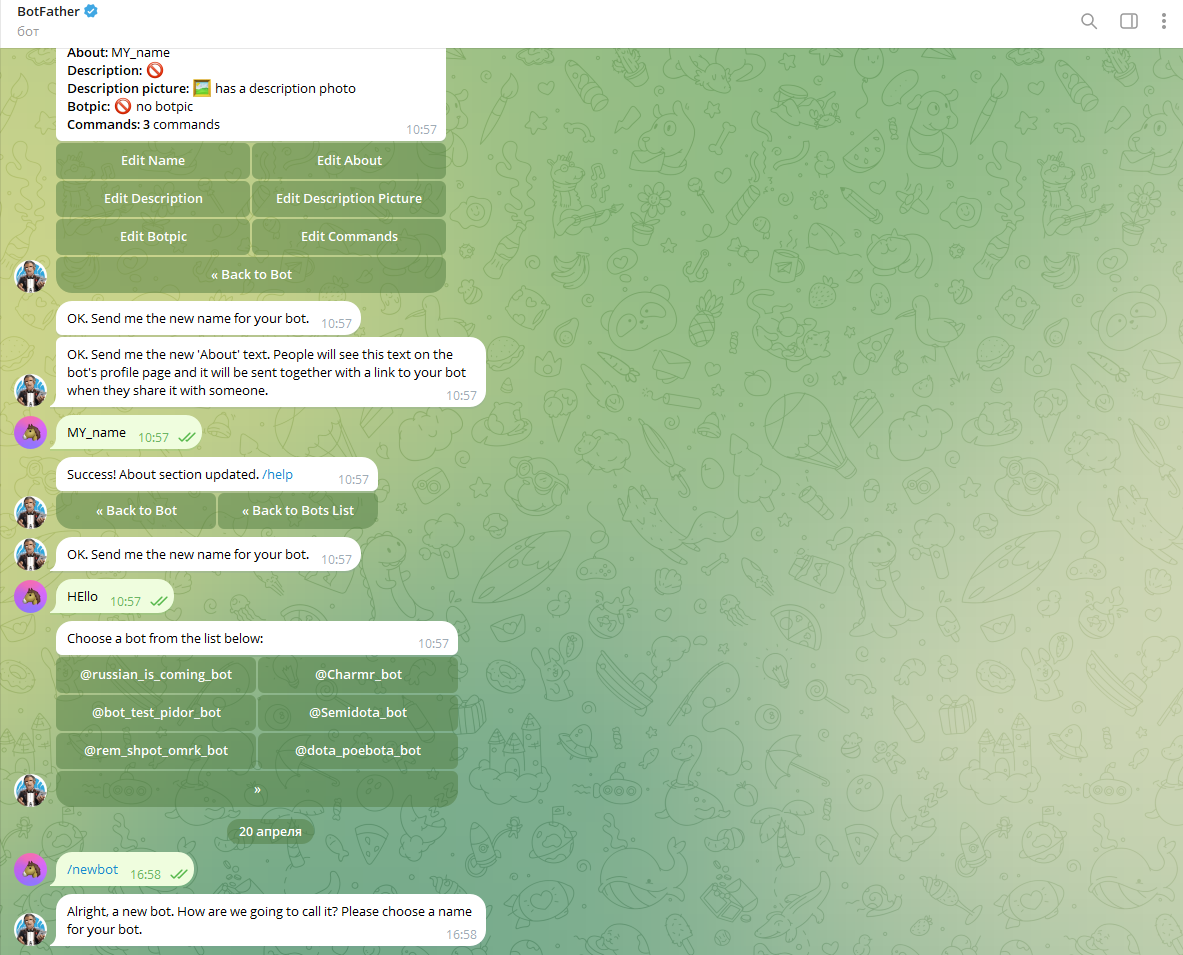


(Команда /start рисунок 8.)

Структура нашего чат бота будет сформировано следящим образом:

1. Главное меню
   1. Смотреть анкеты – переход к просмотру анкет
   2. Редактирование анкеты – переход к редактированию
   3. Магазин – переход к покупке
2. Смотреть анкеты
   1. Лайк – анкета понравилась
   2. Дизлайк – анкета не понравилась
   3. Выход – выйти к главному меню
3. Редактирование анкеты
   1. Имя – поменять имя
   2. Возраст – поменять возраст
   3. Пол – поменять пол
   4. Описание – поменять описание
   5. Фото – поменять фото
4. Магазин
   1. 100 нажатий – покупка просмотров 100 анкет
   2. VIP – бесконечный просмотр и статус в анкете
   3. Регистрация чат-бота.

Первым шагом разработки приложения является регистрация у специального чат-бота «BotFather». Регистрация начинается с команды «/newbot», после чего предлагается ввести название чат-бота с обязательным условием: в конце название должно быть указано «Bot» или «\_bot». Если все условия были удовлетворены, то «BotFather» выдает токен (специальный набор символов для доступа к HTTP API Telegram Bot) и URL-адрес для доступа с чат-боту. Пример регистрации чат-бота представлен на рисунке 9.



(Пример регистрации чат-бота представлен на рисунке 9.)

Для установки дополнительных параметров, таких как иконка чат-бота, приветственное сообщение, описание чат-бота, а также удаление имеющихся чат-боты, существуют следующие команды (таблица 1).

Таблица 1 – Доступные команды для изменения чат-ботов

|  |  |
| --- | --- |
| Команда | Описание |
| /setname | Изменяет существующее имя |
| /setdescription | Присваивает текст, который будет  отображаться при первом открытии чатбота |
| /setabouttext | Присваивает текст в поле «О чат-боте» |
| /setuserpic | Присваивает выбранную картинку |
| /setcommands | Позволяет создать список доступных команд |
| /deletebot | Удаляет выбранного чат-бота |

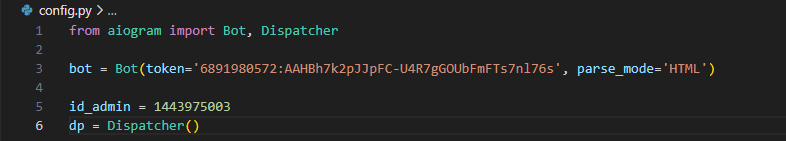
Помимо команд для изменения основных параметров чат-бота существует ряд команд, которые позволяют выводить неизменяемые параметры (токен), а также присваивать значения, которые представлены в таблице 2.

Таблица 2. – Доступные команды для дополнительной настройки чат-бота

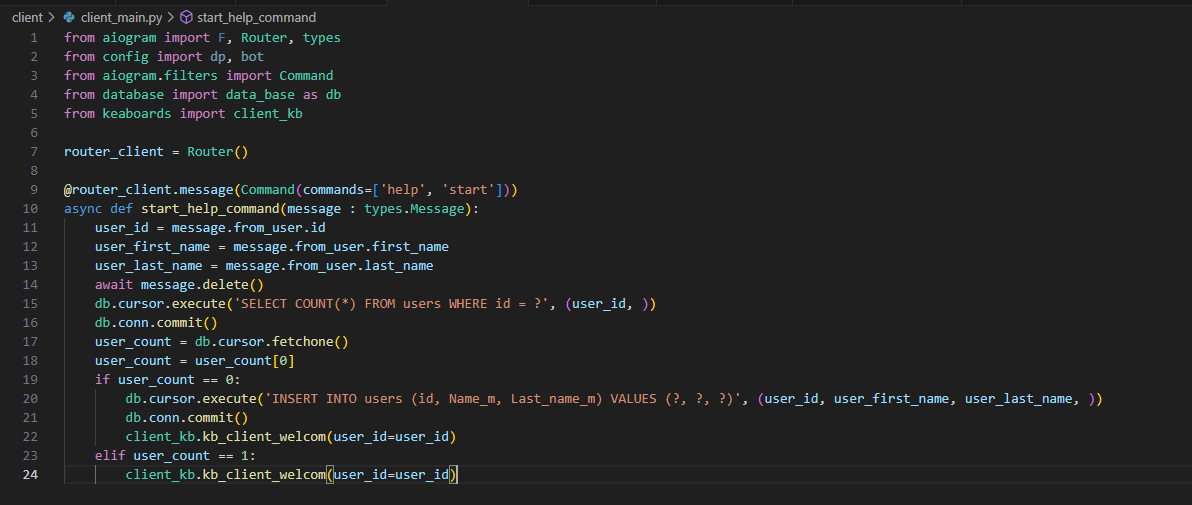
|  |  |
| --- | --- |
| Команда | Описание |
| /token | Возвращает полученный ранее токен у выбранного бота |
| /revoke | Аннулирует токен доступа к боту |
| /setinline | Включает или выключает возможность вызывать бота из других чатов |
| /setinlinegeo | Включает или выключает возможность передавать местоположение бота из другого чата |
| /setinlinefeedback | Позволяет получать информацию о количестве выбранных пользователями команд |
| /setjoingroup | Определяет может ли быть добавлен в групповые диалоги бот |
| /setprivacy | Включает режим конфиденциальности. В этом режиме бот получает, обрабатывает и отсылает обратно информацию отдельно для каждого пользователя в чат |

После настроек на стороне Telegram и получении токена можно приступить к разработке программной части чат-бота.

* 1. Реализация клиентской части чат-бота.
     1. Разработка, создание и обработка

Для начала реализации сформируем файл, куда поместим наш токен и укажем режим разметки, как и описывалась выше у aiogram есть своя документация, к которой можно обратиться в случае возникновения вопросов. Пример создания config – файла изображен на рисунке 10.

(Файл config.py чат-бота рисунке 10.)

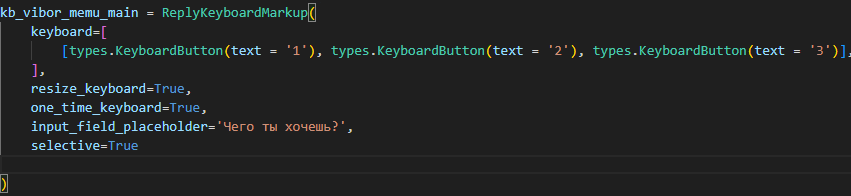
Обращаясь в заранее сформированному шаблону меню и интерфейса, создаем папку куда разместим всю клиентскую обработку. После относительно недавних обновлений, чат-бот не может писать пользователю, пока пользователь первый не напишет. Оценив в пункте 3.2 Целевая аудитория возможность прихода новых пользователей, которые уже имеют знания пользования чат-ботами, необходимо заняться обработкой команды /start и всеми необходимыми командами который ранее были описаны. Пример создания и обработки команды /start и /help на рисунке 11. Вывод команды /start и /help (см. Приложение 2)

(Обработка команды /start и /help, рисунок 11.)

Для создания отправки клавиатуры в aiogram используется объект ReplyKeyboardMarkup который содержит в себе 6 объектов внутри себя (таблица 2).

Таблица 2 – Объекты внутри ReplyKeyboardMarkup

|  |  |  |
| --- | --- | --- |
| Field | Type | Описание |
| text | String | Текст кнопки. Если ни одно из необязательных полей не используется, он будет отправлен в виде сообщения при нажатии кнопки |
| request\_users | KeyboardButtonRequestUsers | Необязательный. Если указано, при нажатии на кнопку откроется список подходящих пользователей. Идентификаторы выбранных пользователей будут отправлены боту в служебном сообщении “users\_shared”. Доступно только в приватных чатах. |
| request\_chat | KeyboardButtonRequestChat | Необязательный. Если указано, при нажатии на кнопку откроется список подходящих чатов. При нажатии на чат бот отправит его идентификатор в служебном сообщении “chat\_shared”. Доступно только в приватных чатах. |
| request\_contact | Boolean | Необязательный. Если значение равно True, номер телефона пользователя будет отправлен в качестве контактного при нажатии кнопки. Доступно только в приватных чатах. |
| request\_location | Boolean | Необязательный. Если значение равно True, при нажатии кнопки будет отправлено текущее местоположение пользователя. Доступно только в приватных чатах. |
| request\_poll | KeyboardButtonPollType | Необязательный. Если указано, пользователю будет предложено создать опрос и отправить его боту при нажатии кнопки. Доступно только в приватных чатах. |
| web\_app | WebAppInfo | Необязательный. Если указано, описанное веб-приложение будет запущено при нажатии кнопки. Веб-приложение сможет отправлять служебное сообщение “web\_app\_data”. Доступно только в приватных чатах. |

Пример реализации клавиатуры одного из меню чат-бота изображен на рисунке 12. Вывод клавиатуры (см. Приложение 3)

(Клавиатура основного меню, рисунок 12.)

После создание главного меню необходимо обработать все переходы по всем выбором, которые были описаны в пункт 3.3. После обработки команды, бот считывает из объекта message другие объекты, которые содержат 80 объектов с различным содержимым из интересующих нас и часто используемых (таблица 3).

Таблица 3 – Объекты внутри message

|  |  |  |
| --- | --- | --- |
| Field | Type | Описание |
| message\_id | Integer | Уникальный идентификатор сообщения в этом чате |
| from\_user.id | Integer | Уникальный идентификатор пользователя |
| from\_user.first\_name | String | Имя указанное пользователем в Telegam |
| from\_user.last\_name | String | Фамилия указанная пользователем в Telegam |
| text | String | Текст отправленный пользователем полученный через message\_id, |
| photo | Array of PhotoSize | Необязательный. Сообщение - это фотография, доступные размеры фотографии |

Следующей задачи после обработке всех переходов из главного меню, необходимо реализовать анкету пользователя. В пункте 3.2 рисунок 8, взятый за шаблон отображение анкеты. 3.5.2

* + 1. Асинхронное выполнение, asyncio

При формировании функции выполнения, задумываешь над ее выполнением, вызовом и выводом результата, но при ее выполнении интегратор выполняет конкретное действие сверху вниз независимо от происходящих действие поверх выполнения функции, что привод к выполнении задачи друг за другом или вовсе ее пропуском. На помощь приходит асинхронное выполнение.

Прежде чем мы углубимся в технические аспекты, важно иметь некоторое базовое понимание терминов, часто используемых в этом контексте.

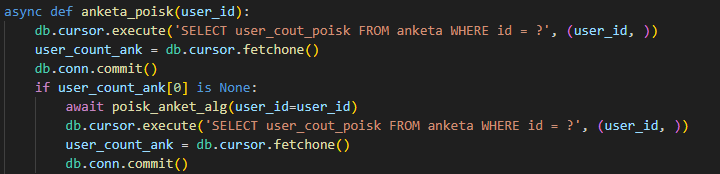
В синхронных операциях задачи выполняются друг за другом. В асинхронных задачи могут запускаться и завершаться независимо друг от друга. Одна асинхронная задача может запускаться и продолжать выполняться, пока выполнение переходит к новой̆ задаче. Асинхронные задачи не блокируют (не заставляют ждать завершения выполнения задачи) операции и обычно выполняются в фоновом режиме.

Конкурентность подразумевает, что две задачи выполняются совместно, в зависимости от нагрузки. Когда мы попросили позвонить друга, а сами писали письмо, то задачи выполнялись параллельно.

Параллелизм по сути является формой̆ конкурентности. Но параллелизм зависит от оборудования. Например, если в CPU только одно ядро, то две задачи не могут выполняться параллельно. Они просто делят процессорное время между собой̆. Тогда это конкурентность, но не параллелизм.

Asyncio предоставляет нам цикл событий наряду с другими крутыми вещами. Цикл событий (event loop) отслеживает события ввода/вывода и переключает задачи, которые готовы и ждут операции ввода/вывода

Идея очень проста. Есть цикл обработки событий. И у нас есть функции, которые выполняют асинхронные операции ввода-вывода. Мы передаем свои функции циклу событий и просим его запустить их для нас. Цикл событий возвращает нам объект Future, словно обещание, что в будущем мы что-то получим. Мы держимся за обещание, время от времени проверяем, имеет ли оно значение (нам очень не терпится), и, наконец, когда значение получено, мы используем его в некоторых других операциях [т.е. мы послали запрос, нам сразу дали билет и сказали ждать, пока придёт результат. Мы периодически проверяем результат и как только он получен мы берем билет и по нему получаем значение, прим. переводчика]. Пример использования asyncio в коде изображен на рисунке 13.



(Работа async в aiogram на рисунке 13.)

* + 1. FSM – машинное состояние

Для получения сообщений от пользователя в библиотеке aiogram реализовано функция FSM. FSM — это механизм, который позволяет вашему боту отслеживать состояние и контекст разговора с каждым пользователем. Он особенно полезен, когда необходимо реализовать диалог в несколько шагов или сохранять данные пользователя во время взаимодействия с ботом.

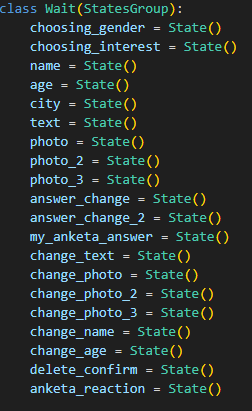
Класс FSMStorage предоставляет способ сохранения и извлечения состояний пользователя. Когда пользователь начинает диалог или выполняет определенные действия, состояние бота может изменяться, и вы можете сохранить эти изменения в хранилище. Затем, когда бот получает новые обновления или команды от пользователя, он может проверить состояние пользователя и определить, какой обработчик использовать или какие данные использовать в контексте текущего состояния. Пример работа FSM в aiogram на рисунке 14.

(Работа FSM в aiogram на рисунке 14.)

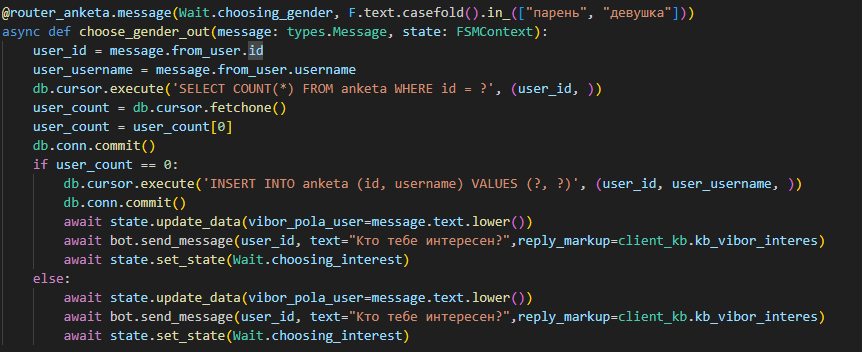
По этой схеме формируем метод и набор функций для формирования нашей анкеты, данные, которые нам будут необходимы:

1. user\_id - Уникальный идентификатор пользователя, именно по этому id будет формироваться, фильтроваться и содержаться в базе данных.
2. last\_name и fist\_name – Как было показано в Таблице 3, необходимая информация для создания анкеты.
3. photo – Объект хранящий уникальный id фото в Telegram
4. opisanie - Объект хранящий в себе текст пользователя
5. age – Объект хранящий в себе число int, возраст указанный пользователем
6. pol – Объект хранящий в себе текст str, указанный пользователем половую принадлежность
7. sity - Объект хранящий в себе текст str, указанный пользователем город в котором будет проходить поиск.

Для перехода в FSM, необходимо сформировать класс, который наследуется от класса StatesGroup и присваиваться новый объект внутри этого класса. Пример реализации класса Wait изображен на рисунке 15.



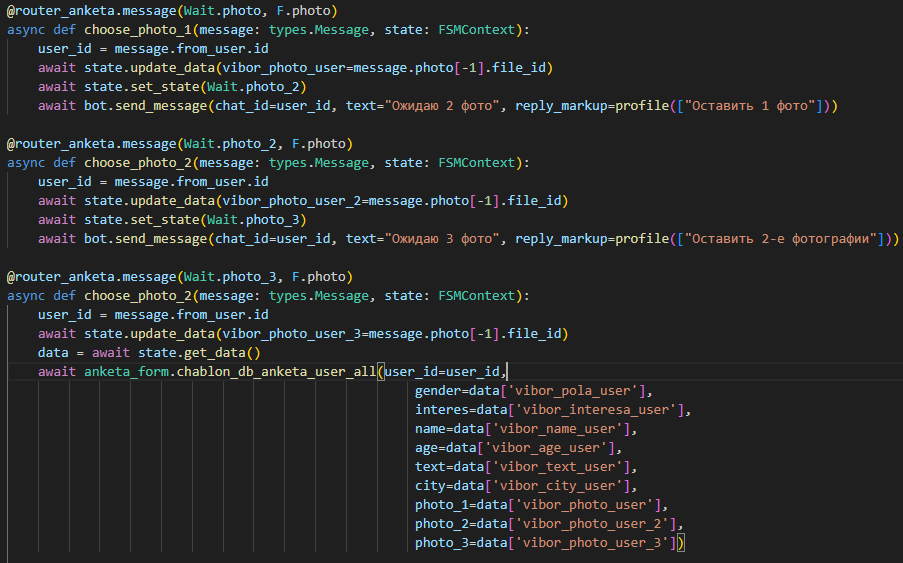
(Класс Wait рисунок 15.)

После формирования класса, необходимо инициализировать начало FSM. В случае нашего проекта, после начальной проверки или попытки перейти куда-либо в главном меню, бот отправляет сообщение с предупреждением о отсутствие анкеты и отправляет клавиатуру с единственной кнопкой “ОКЕЙ”, от которой и формируется начало FSM. По рисунку 13. и всем перечисленным и необходимым данным, формируем функции для получения нужной информации для пользователя. Пример реализации функции перехода, обработки и сохранения переданной информации изображен на рисунке 16.

(Пример функции choose\_gender\_out рисунок 16.)

Все дальнейшие функции формируется единообразно, при реализации функции необходимо учитывать, что ранее пользователь уже мог иметь анкету в проекте и нет необходимости заполнять тужу информацию повторно, что в дальнейшем только упростит пользование чат-ботом. Так же, при формировании функции, нужно понимать, что не все переданное пользователем будет являться правильным. Для этого при переходе от одного FSM к другому, можно реализовать фильтр, пропускающий только те типы объектов или только тот тексь или число, которые необходимо для дальнейшего продолжения функции, в противном случае, необходимо реализовать возвращение к исходной функции с повторным заполнением информации. Так же возможно реализовать фильтр и внутри функции, что будет являться неправильным, по причине излишней громоздкости и ухудшения читаемости кода.

При формировании функции, отвечающей за выбор города, необходимо понимания конкретных городов. Россия страна большая и не хватит экрана для отображения всех возможных городов. Для простоты заполнения будет возможность выбрать только 2 города, в дальнейшем возможно реализовать аналог заполнения как - Camilfo Club.

При обработке функции, отвечающей за получения уникального идентификатора фотографии, можно столкнутся с проблемой обработки получения этого идентификатора. В этом проекте упрощенная схема обработки, которая реализована следующим образом. Пример обработки фотографии изображен на рисунке 17.

(Пример функций обработки рисунок 17.)

После каждой функции, за исключением последней, необходимо реализовать вариант, остановки FSM и возможностью оставить одну либо 2 фотографии, для этого реализуется отдельные функции.

Для сохранения переданной пользователем информации, в методе FSM объявляется метода state.update\_data, куда помещается наименование переменной и медом хранящий в себе информацию об нужном объекте. После получения от пользователя всех необходимых данных и обработке их в соответствующий функциях, методом FSM “state.get\_dat()” получаем в виде словаря набор данных формата ключь – vibor\_photo\_user и значение – уникальный идентификатор (id) фото. Далее значения из словаря помешаются в функцию, которая отправляет весь набор значения в базу данных.

Следующим действием является вывод полученной информации из базы данных уже готового варианта в виде анкеты. Необходимо уточнить, что при выводе анкеты, необходимо уточнить у пользователя, уверен ли он в той информации, которую он передал и дать возможность исправить ее, если такова потребность есть. Работа FSM (см. Приложение 4)

* + 1. Создание Анкеты и работа связанных функций

После заполнения анкеты появляется возможность перейти во все разделы из главного меню. При переходе от заполнения анкеты к главному меню, бот присылает информацию,

При формировании анкеты, незримо для пользователя, но, более чем заметно для разработчика, сразу после выбора половой принадлежности, формируется отдельные переменные, связанные с поиском интересующих анкет, а именно:

1. Возрастной поиск – диапазон поиска по возрасту. Пример – 18/26, где 18 это начальный возраст поиска, а 26 его придел.
2. Поиск по полу – в зависимости от того, что выбор пользователь, формируется противоположное выбору пользователя переменная, которая, аналогичным образом сохраняется.
3. Ссылка – ссылка на профиль пользователя

Для отображения анкеты, должная быть сформирована функция, которая повторная вызывается при ее объявлении. При нажатии кнопки в главном меню отвечающая за переход к просмотрам анкет, реализуется функция, просматривающая базу данных на наличие анкет, по заданным требованиям и ограничениям. Далее идет процесс поиска. Реализации поиска анкет и их обработка изображена на рисунке 18.

(Функция поиска анкет, рисунок 18.)

Поиск формируется на основе информации, заложенной в базе данных, а именно:

* Половая принадлежность
* Возраст
* Город

Из полученной запроса в базу данных, приходит список, дальнейшим содержанием:

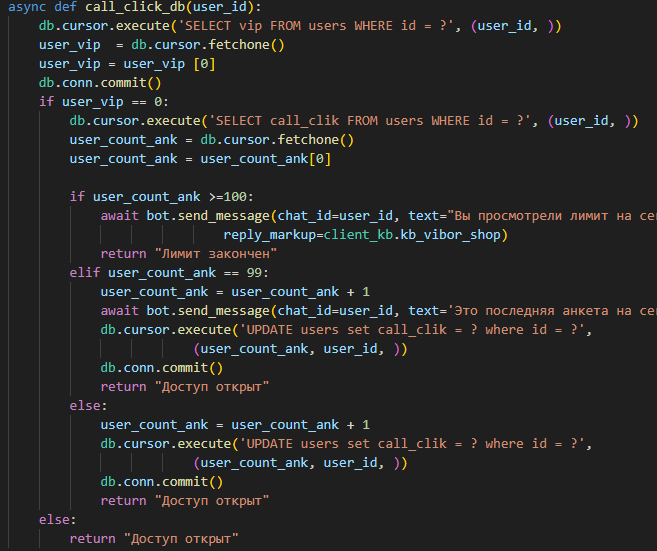
* id – Уникальный идентификатор
* user\_name – Имя пользователя анкеты
* age – Возраст пользователя
* text – Описание анкеты
* photo – Набор id фотографий, их может быть как 1 так и 3.

Сформировать этот список можно и получив его простым запросом и в дальнейшем обработать его внутри функции. После получения списка готовых анкет, он копируется помешается в новую переменную и отправляется в базу данных, без первой анкеты. Анкета, которая была удалена из списка, помешается в функцию, которая выводит полноценную анкету.

После получения пользователем анкеты, предоставляется выбор из 4 пунктов, а именно

* Лайк – Анкета понравилась
* Дизлайк – Анкета не понравилась
* Выйти – Закончить просмотр анкеты, отправляет в главное меню

При нажатии клавиши “Лайк”, происходит запрос в базу данных, к месту, где лежит ячейка с информацией о лайках анкеты, куда помещается id пользователя прожавший эту клавишу. При аналогичном действии происходит запрос в базу данных к ячейке, где лежит информация о наличии списка анкет, если эта ячейка пустая, бот присылает уведомление о том, что список закончился, и будут повторно присылаться эти же анкеты, формируется новым список и повторяется действия выше, если список не пуст, список копируется и отправляется в базу данных без первой анкеты, и действие повторяется. Так же при нажатии, бот формирует запрос в базу данных к ячейке, где содержится информация о количестве нажатий, если количество нажатий превышает порог, заданный разработчиком, бот присылает уведомление о том, что на сегодня, возможность просматривать анкеты запрещено и отправляет сообщение возможность перехода в магазин. Это действие реализовано, сразу после нажатия, с целью проверки. Проверка количества нажатий отражено на рисунке 19.

(Функция проверки, рисунок 19.)

При нажатии клавиши “Дизлайк”, аналогичный процесс проверки, запрос в базу данных к месту, где лежит список анкет, если список пуст, повторяется действия выше, аналогичный процесс, если список не пуст. Информация о Дизлайке нигде не сохраняется, это действие соизмеримо с простым пропуском анкеты.

При нажатии клавиши “Выйти”, бот присылает сообщение с содержанием главного меню.

Реализовав возможность просмотр анкет, необходимо связать понравившиеся анкеты. Спустя заданное разработчиком время, срабатывает функция, которая формирует список людей, которым понравилась анкета для каждого пользователя. Так же происходит проверка, если список пустой, дальнейшие действия не касаются этого пользователя. После сформированного списка пользователей, у которых в ячейках содержится информация, о том, что его анкета понравилась, происходит процесс, уведомления, которое содержит информацию, а именно:

1. Количество пользователей, заинтересовавшихся анкетой
2. Клавиатура, которая содержит данную данные кнопки
   1. Показать анкеты
   2. Продолжить просмотр
   3. Выйти

При нажатии клавиши “показать анкеты”, формируется список из анкет, и аналогичному способу выше, происходит их показ, если анкета понравилась, в этом случае, помимо шаблонного отображения анкеты, вставляется ссылка на Telegram этого пользователя. При аналогичных действиях с кнопкой “Дизлайк”, кнопка “Продолжить просмотр” убирает из списка, данные анкеты.

При нажатии кнопки “Выйти”, бот присылает шаблон связанный с главным меню

Так же, спустя заданное время, происходит процесс, уведомления о возможностях покупки, Vip в магазине, с подробным описанием их возможностей. Работа поиска анкет и ограничение просмотра (см. Приложение 5)

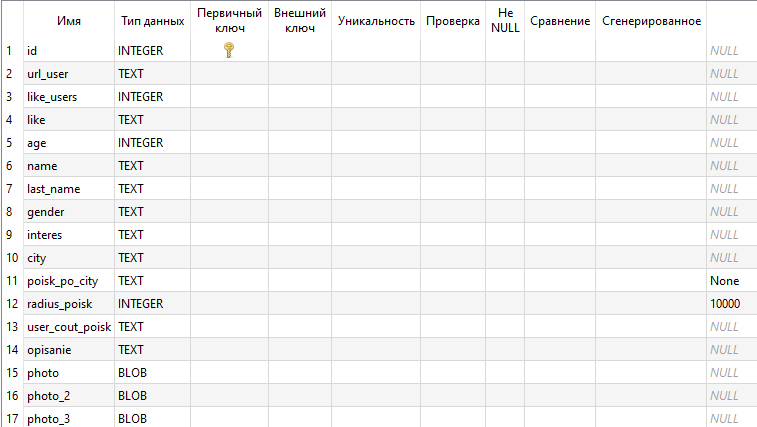
* 1. Проектирование и реализация базы данных чат-бота.

В пункте 3.5 неоднократно упоминалась база данных. В python есть специальная библиотека для создания запросов в базу данных - sqlite3, с ней и будем работать. Структура таблицы users базы данных user\_db.sql изображена на рисунке 20.

(Таблица user, базы данных user\_db.sql рисунок 20.)

Таблица имеет следящие ячейки:

1. id – Уникальный идентификатор пользователя
2. Name\_m – Имея указанное пользователем в Telegram
3. Last\_name\_m – Фамилия указанная пользователем в Telegram
4. Call\_clik – Количество просмотренных анкет
5. Vip – Статус пользования

Таблица anketa в базе данных user\_db.sql, содержит всю информацию о анкете пользователя и имеет следующий вид на рисунке 21.

(Таблица anketa, базы данных user\_db.sql рисунок 21.)

Таблица имеет следящие ячейки:

1. id – Уникальный идентификатор пользователя
2. url\_user – Логин пользователя
3. like\_users – Лайки пользователя анкеты
4. like – Лайки пользователей анкеты
5. age – Возраст указанный пользователем
6. name – Имя указанное пользователем
7. last\_name – Фамилия указанная пользователем
8. gender – Пол выбранный пользователем
9. interes – Пол по которому будет производиться поиск
10. city – Город выбранный пользователем
11. poisk\_po\_city – Город по которому будет производится поиск
12. age\_poisk – Возрастное ограничение по поиску
13. user\_cout\_poisk – Место хранения списка анкет для просмотра
14. opisanie – Описание анкеты
15. photo, photo\_2, photo\_3 – Место хранения id фотографии указанный пользователем

Эти таблица были сформированы следующими запросами:

* + - 1. Таблица users

**CREATE** **TABLE** users **(**

id INTEGER **PRIMARY** **KEY,**

Name\_m TEXT**,**

Last\_name\_m TEXT**,**

call\_click INTEGER**,**

vip INTEGER

**);**

* + - 1. Таблица anketa

**CREATE** **TABLE** anketa **(**

id INTEGER **PRIMARY** **KEY,**

**[like]** TEXT**,**

age INTEGER**,**

name TEXT**,**

last\_name TEXT**,**

gender TEXT**,**

interes TEXT**,**

city TEXT**,**

poisk\_po\_city TEXT **DEFAULT** **None,**

age\_poisk TEXT **DEFAULT** **None,**

radius\_poisk INTEGER **DEFAULT** **(**10000**),**

user\_cout\_poisk TEXT**,**

opisanie TEXT**,**

photo BLOB**,**

photo\_2 BLOB**,**

photo\_3 BLOB

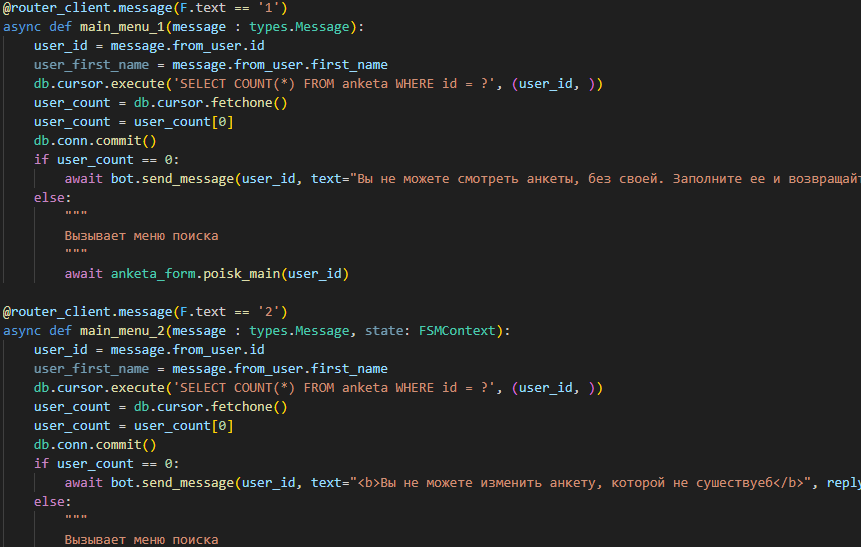
**);**

* 1. Реализация серверной части чат-бота.

Серверная часть чат-бота на платформе Telegram должна решать следующие задачи:

* получать и распознавать сообщение, отправленное пользователем;
* извлекать и сравнивать полученную информацию для составления корректного ответа;
* извлекать информацию из базы данных путем выполнения запросов с полученными параметрами;
* составлять и отправлять ответ в виде текста, сформированного на основе информации пользователя.

Для того чтобы чат-бот имел возможность распознавать сообщение пользователя каждая кнопка клавиатуры имеет так называемую обратную информацию. После нажатия кнопки пользователем серверная часть сравнивает полученный ответ со значением или массивом значений. Пример обработки нажатия кнопки “1” в главном меню имеет следующий вид на рисунке 22.



(Обработка нажатия кнопки “1” рисунок 22.)

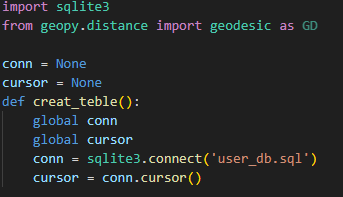
Обратившись к официальной документации, выстраивается понимания обработки входящих действий и объявления необходимых функций и классов.

При реализации временного исполнения функций, существует библиотека apscheduler. Для ее использования необходимо ее предварительное объявление с указание “Тайм зоны”. С следующих действиях, после последующих объявлений, указывается функция, которая должна исполняться, метод по которому будет происходить определение интервала, ограничение при их необходимости связанный с будними или выходными днями, часовой и минутный триггер, время по которому будет происходить выполнение функции или кода. Пример объявления и формирования временного исполнения функции имеет следующий вид на рисунке 23.

(Планировщик задач рисунок 23.)

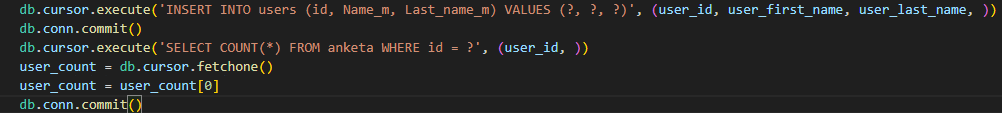
Так же, при необходимости возможно реализация интервального исполнения кода, для этого в переменную trigger помешается в формате str, текст “interval”. В следующих объявлениях указывается интервал исполнения. Работа функций в указанное время (см. Приложение 7)

При извлечении информации из базы данных или ее передачи, в python реализован следящий метод. Объявляется 2 глобальные переменные, куда помещается подключение к базе данных и ее курсор. Пример формирования подключение к базе данных, расположенной в корне проекта, имеет следующий вид на рисунке 24.



(Файл data\_base.py рисунок 24.)

При передаче информации в базу данных используется стандартный язык SQL, и используется уже объявленные глобальные переменные. Пример использования метода передачи информации в базу данных имеет следующий вид на рисунке 25.



(Передача и получения данных в базу данных рисунок 25.)

* + 1. Реализация магазина.

Для принимать платежи в Telegram. необходимо перейти в @BotFather, выбрать своего тестового бота и подключить к нему эквайринг, в нашем случае тестовый Token и карту для обработки платежей. После получения необходимых данных формируем счет на оплату. В aiogram.types есть классы LabeledPrice и PreCheckoutQuery. Коротко о них

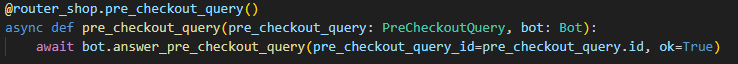
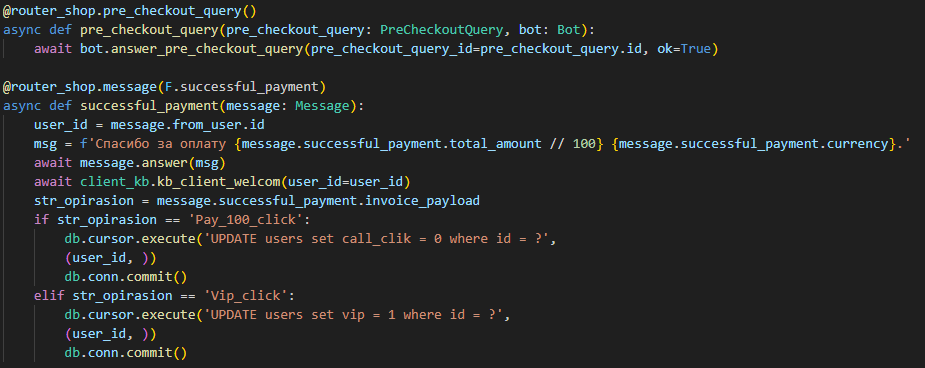
1. LabeledPrice - Этот объект представляет собой часть цены на товары или услуги. Объекты этого класса сравнимы с точки зрения равенства. Два объекта этого класса считаются равными, если их labelи amount равны. Позволяет формировать цену на товар. Ключевые поля :
   1. Label – Описание
   2. Amount – цена товара
2. PreCheckoutQuery - Этот объект содержит информацию о входящем запросе предварительного оформления заказа. Объекты этого класса сравнимы с точки зрения равенства. Два объекта этого класса считаются равными, если они idравны. Формирует ответный запрос на сервера Telegram для подтверждения оплаты. Ключевой факт, ответ должен поступить в течении 10 секунд, в противном случае, запрос на оплату считается недействительным и отклоняется. Ключевые поля –
   1. Id - Уникальный идентификатор запроса.
   2. from\_user - Пользователь, отправивший запрос.
   3. Currency - трехбуквенный код валюты ISO 4217.
   4. total\_amount - Общая цена в наименьших единицах валюты (целое, а не плавающее/двойное).
   5. invoice\_payload - Указанная ботом полезная нагрузка счета-фактуры.
   6. shipping\_option\_id - Идентификатор выбранного пользователем варианта доставки.
   7. order\_info - Информация о заказе, предоставленная пользователем.

Для формирования счета объявляется асинхронная функция и удобный способ ее инициализации в нашем случаем через Роутер. В классе bot есть метод send\_invoice, который содержит 28 переменных, которые имеют ключевую необходимость, из интересующих нас:

1. chat\_id – Объект с информации о отправителе
2. title – Название объекта покупки
3. description – Описание объекта покупки
4. payload – Call\_back информация после формирования и оплаты счета. Необходим для обработки последующих действий связанные с обращением к функции оплаты
5. provider\_token – Token провайдера обрабатывающий счет на оплату
6. currency – Наименование валюты
7. prices – Цена товара
   1. Возможно реализовать формировании цены типа НДС, Бонусы и Скидки
8. photo\_url – Ссылка на фото изображения

Пример формирования асинхронной функции оплаты отражен в Приложение 1

После сформированного счета, необходимо ответить серверам Telegram о готовности обработать сформированный счет. Пример формирования ответа изображён на рисунке 26. Формирования расчётного счета (см. Приложение 8)



(Асинхронная функция ответа рисунок 26.)

* 1. Размещение чат-бота на удаленном сервере

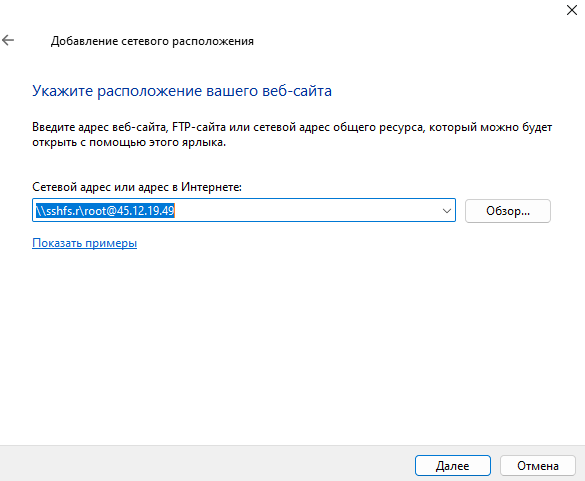
Для постоянного получения доступа к чат-боту, после реализации его необходимо разместить на удаленном сервере, виртуальной машины. В качестве место размещения VPS был выбран Beget.com.

Причина выбора, данного хостинг-провайдера являться следующие пункты:

* Упрощенная форма регистрации
* Формат предлагаемых услуг
* Цены и тарифы
* Расположения серверов

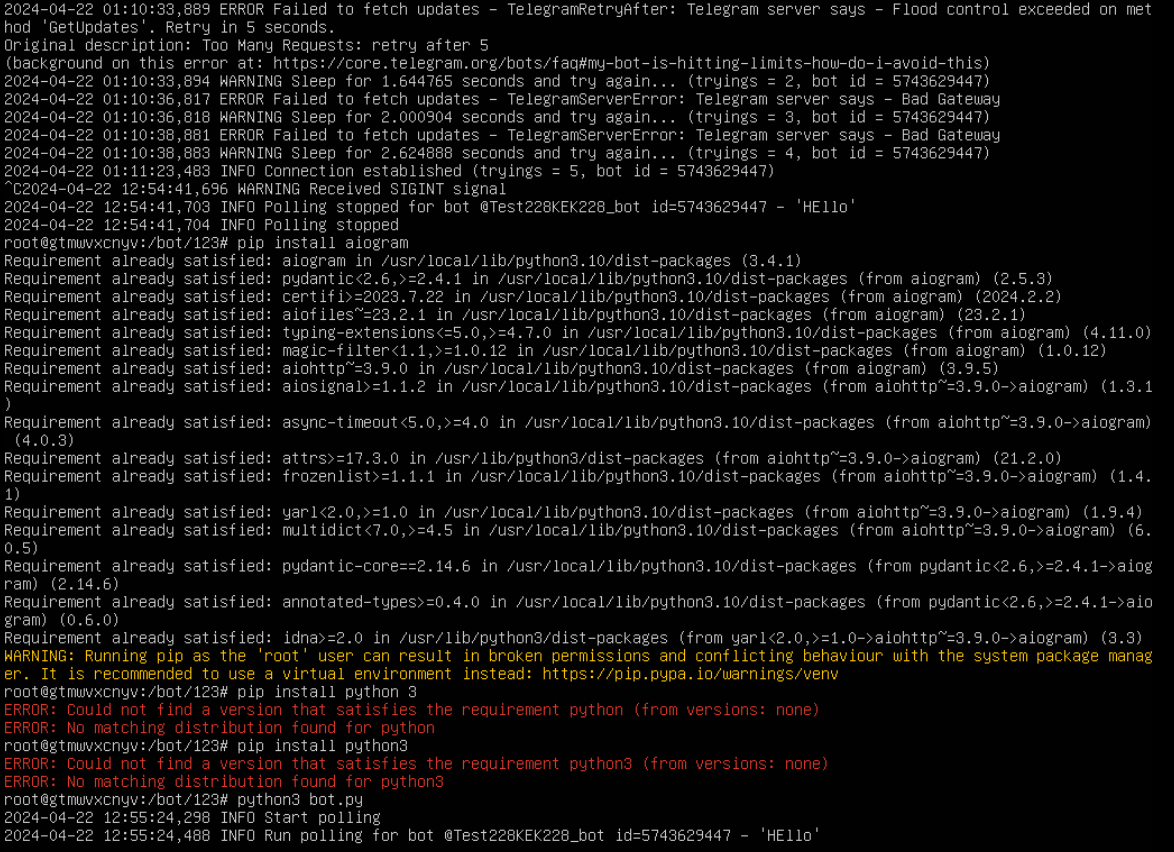
Для размещения чат-бота необходимо пройти регистрацию на платформе. Пример поля регистрации изображен на

После успешной регистрации и успешной захода в личный кабинет, будет сформировано стандартная для всех хостинг провайдеров, главная страница. После пополнения баланса личного кабинета, появляется возможность зайти в файловый менеджер. Для упрощения работа с файловым менеджером, выбранного хостинг-провайдера, можно установить соединение через всем известный FileZilla или подключить сетевой диск на личной десктопном компьютере. Из-за отсутствия FTP подключение к файловому менеджеру будет осуществляться через SSH для этого необходимо установить SSHFS и WinFsp на устройство, с которого будет происходить подключение. Пример подключения к файловому менеджеру изображен на рисунке 27.



(Подключение к файловому менеджеру рисунок 27.)

После успешного подключение необходимо разместить папку с проектом в корень развернутого сервера.

Для запуска проекта, необходимо подключиться через SSH к виртуальному серверу или воспользоваться готовый решением у хостинг-провайдера, а именно терминалом VNS. После успешного подключение необходимо пройти верификацию или указать при подключение логин и пароль VPS, который был заранее сформирован. При успешной верификации необходимо установить все необходимые библиотеки и необходимую версию языка программирования. Необходимо учитывать, что корневая система VPS является Linux, что давит лишь одно действие, для установки всего необходимого. Пример установки и включения чат-бота изображен на рисунке 28.

(Установка и включение чат-бота рисунок 28.)

**ЗАКЛЮЧЕНИЕ**

В настоящее время популярность мессенджеров как средств общения неизменно растет. Компании, семьи, друзья ежедневно пользуются возможностями обмена сообщениями и медиаконтентом на расстоянии. Так же стоит отметить рост популярности такого вида программных продуктов как чат-боты, которые работают на платформах мессенджеров.

Круглосуточная служба поддержки пользователей, конвертирование документов и медиафайлов, заказ такси, поиск необходимых данных и многое другое в настоящее время может быть реализовано в рамках лишь одного мессенджера. Пользователям не придется скачивать множество приложений для решения узконаправленных задач, потому что достаточно иметь лишь мессенджер и необходимый набор чат-ботов, которые не занимают место в памяти смартфона.

В рамках дипломного проекта были выполнены поставленные задачи. Во-первых, были изучены мессенджеры. Было проведено сравнение и анализ достоинств и недостатков, вследствие чего был выбран мессенджер Telegram как самый удобный и доступный в плане документации и возможностей aiogram.

Во-вторых, были изучены имеющиеся аналоги чат-бота на платформе Telegram, а также выявлены их достоинства, недостатки и интересные решения. На основе этого были выявлены требования для разработки авторского чат-бота посвященного чат-ботам для знакомства.

В рамках последней выполненной задачей были выбраны технологии и среда для разработки чат-бота среди которых MySQL, Python, Aiogram и VS Code.

Таким образом, результатом дипломного проекта является полностью реализованный чат-бот для знакомства в Telegram. Код чат-бота (см. Приложение 9)

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1) Messina, C. 2016 will be the year of conversational commerce [Электронный ресурс] / C. Messina // Medium. – 2016. – Режим доступа: https://medium.com/chris-messina/2016-will-be-the-year-ofconversationalcommerce-1586e85e3991

2) Aiogram [Электронный ресурс] : Telegram Documents. – Режим доступа: https://docs.aiogram.dev/en/latest/

3) Аванесян Н. Л., Telegram, как пример мессенджера: возможности и перспективы развития. [Электронный ресурс] / Н. Л. Аванесян // Научныйпотенциал XXI века. – 2017. – Режим доступа: https://elibrary.ru/download/elibrary\_29653726\_34734017.pdf

4) Агальцов, В. Базы данных: В 2-х книгах. Книга 2: Распределенные и удаленные базы данных: Учебник / В. Агальцов – Москва: Форум, 2018. – 271 с.

5) Официальный сайт MySQL [Электронный ресурс]: About – Режим доступа: https://www.mysql.com

6) Официальный сайт Telegam [Электронный ресурс]: API – Режим доступа: https://web.telegram.org/k/.

7) Официальный сайт Python [Электронный ресурс]: API – Режим доступа: https://www.python.org

8) Рейтинг месседжеров 2018 [Электронный ресурс] : Сервис сравнения и выбора приложений и сервисов в сфере IT. – 2018. – Режим доступа: https://coba.tools/compilation/reiting-messendzherov-2018

9) Холмс, С. Стек MEAN. Mongo, Express, Angular, Node. : Учебное пособие / С. Холмс – Санкт-Петербург : Питер Пресс, 2017. – 496 с.

9) vc.ru [Электронный ресурс]: Крупнейшая в рунете площадка для предпринимателей нового поколения. – Режим доступа: https://coba.tools/compilation/reiting-messendzherov-2018

10) Beget [Электронный ресурс]: Хостинг-провайдер – Режим доступа: https://beget.com/ru

**ПРИЛОЖЕНИЕ**

**Приложение № 1**

Асинхронная функция формирования счет на оплату

*@router\_shop***.***message***(**F**.***text* **==** '100 нажатий'**)**

**async** **def** order**(**message**:** Message**,** bot**:** Bot**):**

**await** bot**.***send\_invoice***(**

chat\_id **=** message**.***from\_user***.id,**

title**=** '100 нажатий'**,**

description**=**'Сбрасывает нажатие до 0'**,**

payload**=**'Pay\_100\_click'**,**

provider\_token **=** '381764678:TEST:83693'**,**

currency**=**'rub'**,**

prices**=[**

LabeledPrice**(**

label**=**'Доступ к секретной информации'**,**

amount**=**30000

**),**

LabeledPrice**(**

label**=**'НДС'**,**

amount**=**20000

**),**

LabeledPrice**(**

label**=**'Скидка'**,**

amount**=-**1000

**),**

LabeledPrice**(**

label**=**'Бонус'**,**

amount**=-**40000

**)**

**],**

max\_tip\_amount**=**50000**,**

suggested\_tip\_amounts**=[**1000**,**2000**,**3000**,**4000**],**

start\_parameter**=**'bot'**,**

provider\_data**=None,**

photo\_url**=**'https://play-lh.googleusercontent.com/LHZ34g6xvzFL5RtsJ\_o3hCeNSoYq7OJ3Hw2WsRhrM00SJIZDftNpELElEIVWbXyLpQ=w1024-h500'**,**

photo\_size**=**100**,**

photo\_width**=**800**,**

photo\_height**=**450**,**

need\_name**=True,**

need\_phone\_number**=True,**

need\_email**=True,**

need\_shipping\_address**=False,**

send\_phone\_number\_to\_provider**=False,**

send\_email\_to\_provider**=False,**

is\_flexible**=False,**

disable\_notification**=False,**

protect\_content**=False,**

reply\_to\_message\_id**=None,**

allow\_sending\_without\_reply**=True,**

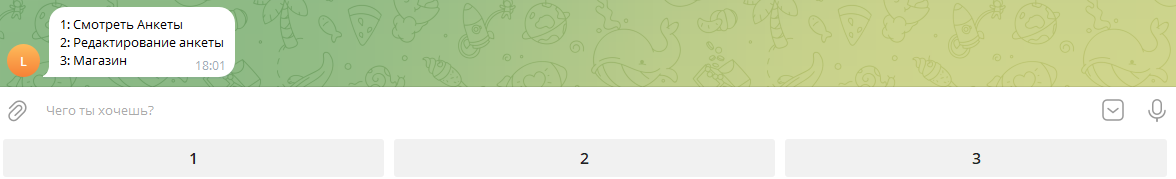
reply\_markup**=None,**

request\_timeout**=None**

**)**

**Приложение № 2**

Вывод /start / help



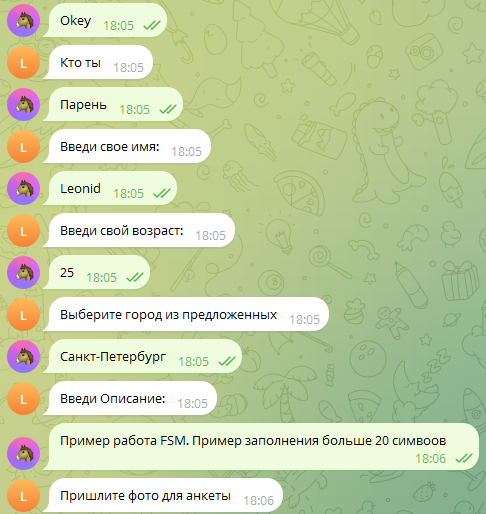
**Приложение № 3**

Вывод клавиатуры



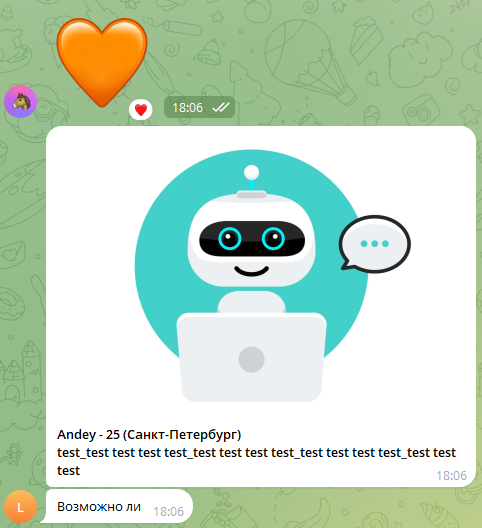
**Приложение № 4**

Работа FSM



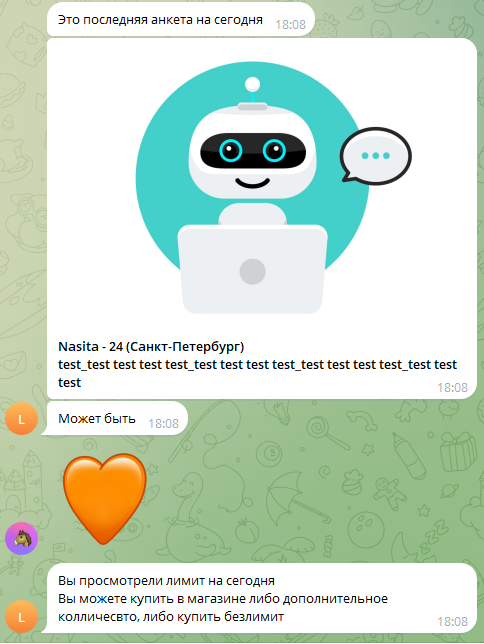
**Приложение № 5**

Поиск и вывод анкет



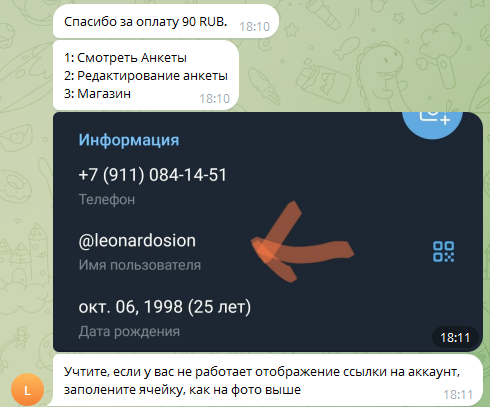
**Приложение № 6**

Ограничение просмотра анкет



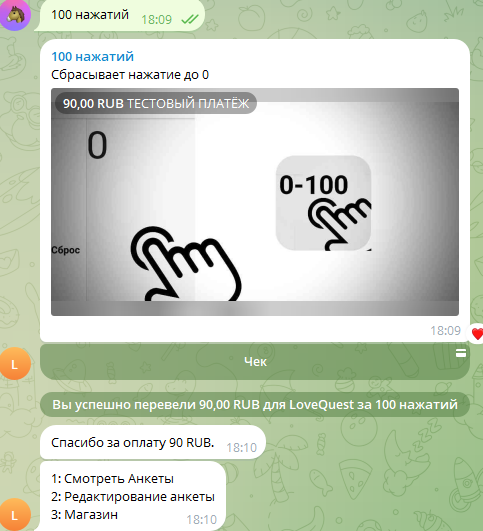
**Приложение № 7**

Работа планировщика



**Приложение № 8**

Работа оплаты



**Приложение № 9**

Код и ссылка чат-бота

<https://github.com/LeonidSila/GB_DIPLOM/tree/main/bot>

https://t.me/Charmr\_bot